Kapitel 3:
Planung von Softwareprojekten

3.1 Terminplanung

3.2 Methoden der Aufwandsschätzung

3.3 Kostenplanung

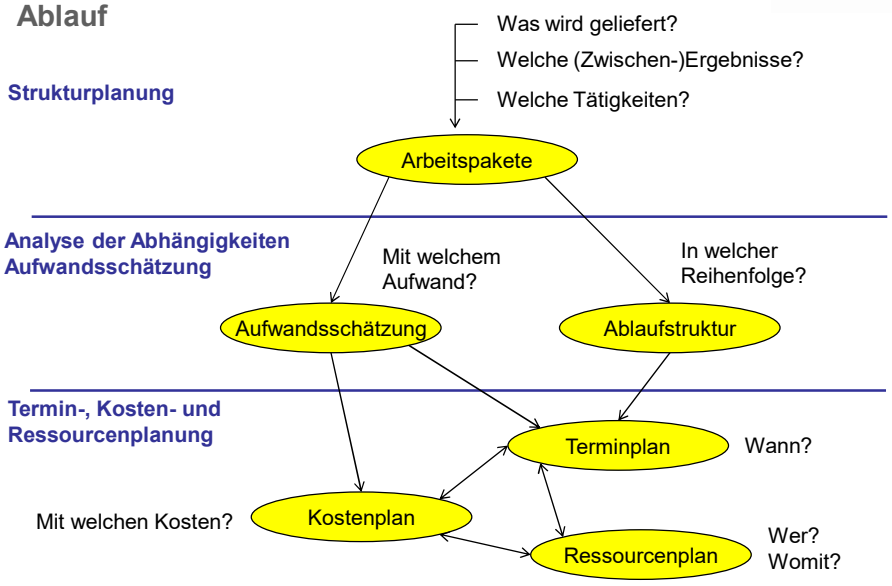
3.4 Ressourcenplanung

3.5 Inhalte eines Projektplans

Projektplanung: Voraussetzungen

- Projektziele festgelegt
- Organisationsform des Projekts festgelegt (Aufbau-Organisation)
- geeigneter Entwicklungsprozess festgelegt (Ablauf-Organisation)
- geeignete Software-Technologien ausgewählt.

Projektplanung
Ablauf



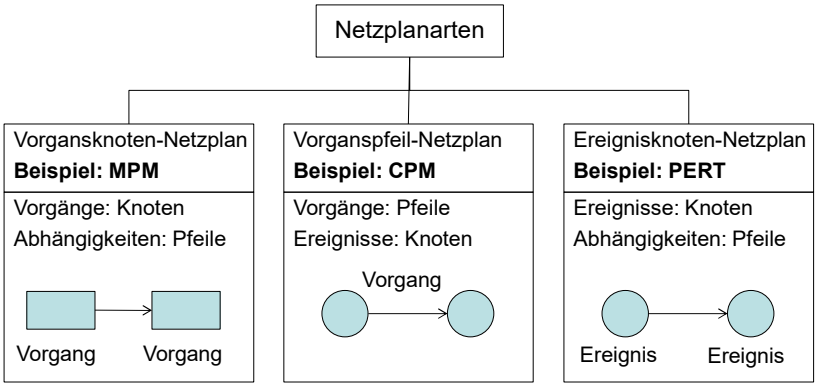
„Gute“ Arbeitspakete ...

- starten bei einem konkreten Ereignis oder Meilenstein,
- enden mit einem konkreten Ergebnis oder Meilenstein,
- können separat abgeschätzt und verfolgt werden,
- sind klein genug, um den Projektstatus transparent zu machen,
- sind groß genug, so dass eine separate Verfolgung sinnvoll ist.

Terminplanung am Beispiel
Projektierung einer Anlage (i)

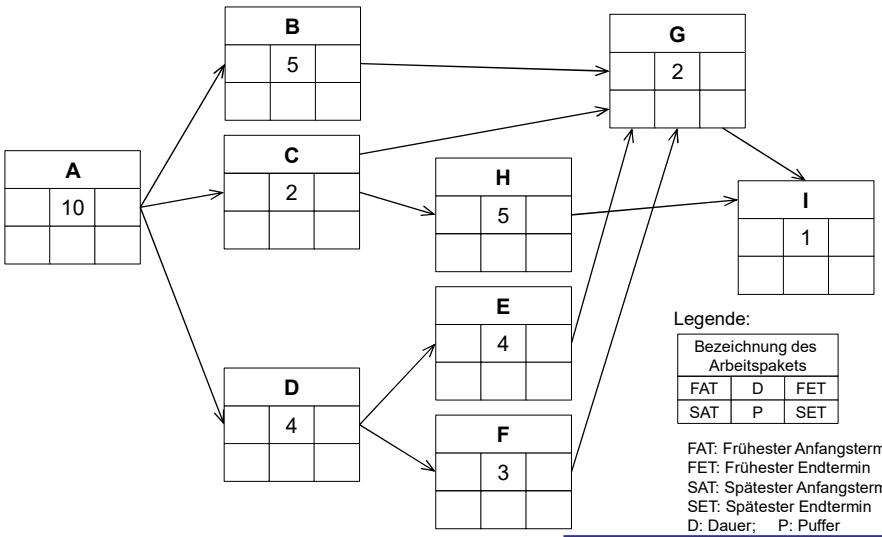
Arbeitspakete		Dauer [Wochen]	Abhängig von
A	Entwurf	10	-
B	Fertigung der Zentraleinheit	5	A
C	Bereitstellung der Ein- und Ausgabegeräte	2	A
D	Projektierung der Software	4	A
E	Projektierung der Testprogramme	4	D
F	Erstellung kundenspezifischer Programme und Testprogramme	3	D
G	Anlagentest	2	B, C, E, F
H	Bereitstellung der Anschluss-Hardware	5	C
I	Installation	1	H, G

Aufbau eines Terminplans
Netzplan



MPM: Metra Potential Method
CPM: Critical Path Method
PERT: Program Evaluation and Review Technique

Terminplanung am Beispiel
Projektierung einer Anlage (ii)



Legende:

Bezeichnung des Arbeitspakets		
FAT	D	FET
SAT	P	SET

FAT: Frühester Anfangstermin
FET: Frühester Endtermin
SAT: Spätester Anfangstermin
SET: Spätester Endtermin
D: Dauer; P: Puffer

Abhängigkeiten der Arbeitspakete

Mögliche Abhängigkeiten zwischen zwei Arbeitspaketen A und B:

- Normalfolge (Ende-Anfangs-Folge)
  Anfang des Arbeitspakets B hängt vom Ende des Arbeitspakets A ab.
  Formale Notation: [A] EA [B]

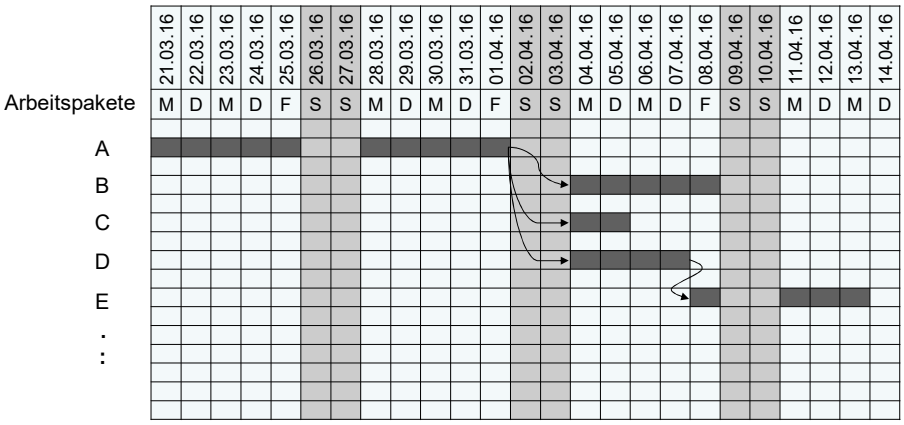
- Anfangsfolge (Anfangs-Anfangs-Folge)
  Anfang des Arbeitspakets B hängt vom Anfang des Arbeitspakets A ab.
  Formale Notation: [A] AA [B]

- Endfolge (Ende-Ende-Folge)
  Ende des Arbeitspakets B hängt vom Ende des Arbeitspakets A ab.
  Formale Notation: [A] EE [B]

- Sprungfolge (Anfangs-Ende-Folge)
  Ende des Arbeitspakets B hängt vom Anfang des Arbeitspakets A ab.
  Formale Notation: [A] AE [B]

Aufbau eines Terminplans
Gantt-Diagramm

Gantt-Diagramme sind Auswertungen von Netzplänen in Form von Balkendiagrammen.



Kapitel 3:
Planung von Softwareprojekten

3.1 Terminplanung

3.2 Methoden der Aufwandsschätzung

3.3 Kostenplanung

3.4 Ressourcenplanung

3.5 Inhalte eines Projektplans

Methoden der Aufwandsschätzung
Basismethoden (i)

Vergleichsmethoden

- Analogiemethode: Erfahrungsbasierter Vergleich der zu schätzenden Entwicklung mit bereits abgeschlossenen ähnlichen Entwicklungen.

- Relationsmethode: Die zu schätzende Entwicklung wird anhand von bestimmten Kriterien mit ähnlichen Entwicklungen direkt verglichen (in Relation gesetzt).

Vergleichsmethoden sind für eine erste grobe Schätzung gut geeignet.

Methoden der Aufwandsschätzung
Basismethoden (ii)

Kennzahlenmethoden

- Prozentsatzmethode: Aus abgeschlossenen ähnlichen Projekten ist bekannt, wie sich der Aufwand auf die einzelnen Entwicklungsphasen verteilt hat. Auf Basis dieser Verteilung wird der Aufwand der neuen Entwicklung geschätzt.

- Multiplikatormethode: Das zu entwickelnde System wird soweit in Teilsysteme zerlegt, bis jedem Teilsystem ein seinem Umfang entsprechender Aufwand zugeordnet werden kann.

Die Anwendung von Kennzahlenmethoden ist problematisch. Warum?

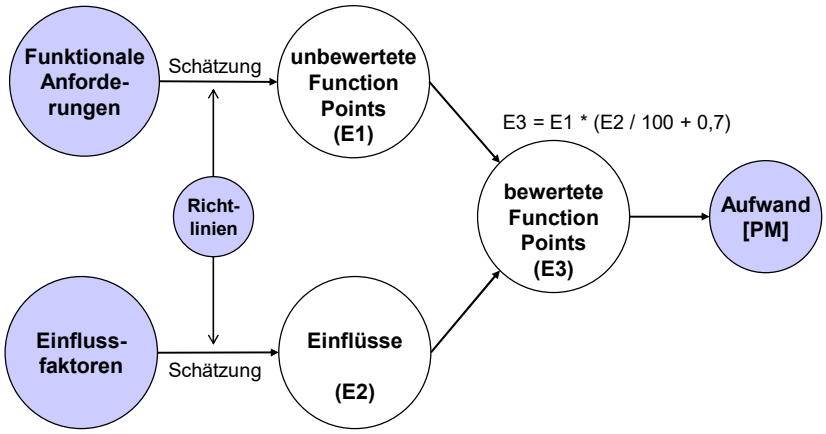
Methoden der Aufwandsschätzung
Algorithmische Methoden

- Function Points
- COCOMO II (Constructive Cost Model)
- Use Case Points

Keine Schätzmethode: Story Points

- Die Methode wurde für agile Entwicklung definiert,
- Story Points beschreiben die relative Größe einer User Story: A ist großer als B, B ist größer als C, ...
  Wie ist aber diese Größe definiert?
- Aufwände werden mit Story Points nicht geschätzt.

Function Points-Methode (i)



Legende:
PM: Personenmonat

Functio

Kategorie	Anzahl	Klassifizierung	Gewichtung	Zeilensumme
Eingabedaten		einfach	x 3	=
		mittel	x 4	=
		komplex	x 6	=
Abfragen		einfach	x 3	=
		mittel	x 4	=
		komplex	x 6	=
Ausgaben		einfach	x 4	=
		mittel	x 5	=
		komplex	x 7	=
Datenbestände		einfach	x 7	=
		mittel	x 10	=
		komplex	x 15	=
Referenzdaten		einfach	x 5	=
		mittel	x 7	=
		komplex	x 10	=
Summe			E 1	=
Einflussfaktoren (ändern den *Function Point*-Wert um ± 30%)		1 Verflechtung mit anderen Anwendungssystemen (0-5)		=
		2 Dezentrale Daten, dezentrale Verarbeitung (0-5)		=
		3 Transaktionsrate (0-5)		=
		4 Verarbeitungslogik		
		a Rechenoperationen (0-10)		=
		b Kontrollverfahren (0-5)		=
		c Ausnahmeregelungen (0-10)		=
		d Logik (0-5)		=
		5 Wiederverwendbarkeit (0-5)		=
		6 Datenbestands-Konvertierungen (0-5)		=
		7 Anpassbarkeit (0-5)		=
Summe der 7 Einflüsse		E2		=
Faktor Einflussbewertung = E2 / 100 + 0,7		E3		=
Bewertete *Function Points*: E1 * E3				=

Quelle: /IBM 85, S. 12/

COCOMO II
Idee

- Große Software-Projekte kosten überproportional mehr als kleine
- Die Entwicklung komplexer Software kostet überproportional mehr als einfache.
- Es ist sehr schwer, den Aufwand für die Erstellung von Software abzuschätzen, ohne etwas über den Ergebnisumfang und die Komplexität der Aufgabenstellung zu wissen.

→ Erst Ergebnisumfang und Komplexität abschätzen und dann den Aufwand ableiten.

Arbeitspaket
↓
Ergebnisumfang, Komplexität
↓
Aufwand

COCOMO II
Grundformel

Aufwand:

$$\text{Aufwand [PM]} = \text{m} * (\text{Umfang [KDSI]})^{\text{n}}$$

Einfache Projekte (Batchsysteme): m = 2,4, n = 1,05
Mittelschwere Projekte (Dialogsysteme): m = 3, n = 1,12
Komplexe Projekte: m = 3,6, n = 1,2

(Optimale) Projektdauer (Projektlaufzeit, Entwicklungszeit)

$$\text{Projektdauer [M]} = 2{,}5 * (\text{Aufwand[PM]})^{\text{s}}$$

s = 0,4 für Standardsoftware
s = 0,38 für Stapelverarbeitungssysteme (Batchsysteme);
s = 0,35 für Dialogsysteme/Onlinesysteme;
s = 0,32 für Echtzeitsysteme

Teamgröße

$$Teamgröße[P] = \frac{Aufwand\ [PM]}{Projektdauer\ [M]}$$

Legende: M: Monat; PM: Personenmonat; P: Anzahl Projektmitarbeiter; KDSI: Kilo Delivered Instructions

Use Case Points (UCP)

- UCP-Methode wurde 1993 von Gustav Karner unter Betreuung von Ivar Jacobson entwickelt.
- UCP-Methode geht aus der Function Points-Methode hervor.
- Erst 10 Jahre später wurde die Methode von mehreren Unternehmen (IBM, Ericsson, Credit Suisse und vor allem Capgemini sd&m) aufgegriffen und weiterentwickelt.
- Es gibt bereits einige Tools, die die UCP-Zählweise integriert haben.

Als Zählmaß werden bewertet:

- Anzahl Use Cases
- Anzahl Akteure
- Technische Faktoren (nichtfunktionale Anforderungen)
- Projektumgebungsfaktoren

Use Case Points (UCP)
Zählmaß Use Cases

Die Use Cases werden anhand von Komplexität der Transaktionen (Schritte im Ablauf) in drei Kategorien eingeteilt:

- Ein Use Case mit einem Umfang von einer bis drei Transaktionen (inklusive Alternativabläufe) ist „einfach“ und bekommt ein Gewicht W_i von 5 UCP.
- Ein Use Case mit einem Umfang von vier bis sieben Transaktionen (inklusive Alternativabläufe) ist „mittel“ und bekommt ein Gewicht W_i von 10 UCP.
- Ein Use Case mit einem Umfang von mehr als sieben Transaktionen (inklusive Alternativabläufe) ist „komplex“ und erhält ein Gewicht W_i von 15 UCP.

Die Summe der Gewichte W_i aller Use Cases ergibt die Unadjusted Use Case Weights (UUCW).

Use Case Points (UCP)
Zählmaß Akteure

Die Akteure werden ebenfalls in drei Kategorien eingeteilt

- Akteure, die ein anderes System repräsentieren und über eine API mit dem System kommunizieren, werden als einfach betrachtet und bekommen je einen UCP.
- Akteure, die über ein Protokoll (z.B. TCP/IP) mit dem System gebunden sind sowie menschliche Akteure, die direkt über Command-Line Interface mit dem System interagieren, werden der Kategorie mittel zugeordnet und bekommen je zwei UCP.
- Akteure, die eine Benutzerschnittstelle (GUI) nutzen, um mit dem System zu kommunizieren, werden als komplex eingestuft und bekommen je drei UCP.

Die Summe der Gewichte W_i aller Akteure ergibt die Unadjusted Actor Weights (UAW).

Unadjusted Use Case Points: UUCP = UUCW + UAW

Use Case Points (UCP)
Zählmaß Technische und Projektumgebungsfaktoren (i)

Die UUCP werden nun durch Technical Complexity Factor (TCF) und Environmental Factor (EF) korrigiert.

- Einflussgrößen des TCF:

T_i	Einflussgröße	Gewicht W_i
T_1	Distributed systems	2
T_2	Application performance objectives, in either response or throughput	1
T_3	End user efficiency (on-line)	1
T_4	Complex internal processing	1
T_5	Reusability, the code must be able to be reused in other applications	1
T_6	Installation ease	0,5
T_7	Operational ease, usability	0,5
T_8	Portability	2
T_9	Changeability	1
T_{10}	Concurrency	1
T_{11}	Special security features	1
T_{12}	Provide direct access for third parties	1
T_{13}	Special user training facilities	1

Use Case Points (UCP)
Zählmaß Technische und Projektumgebungsfaktoren (ii)

- Die jeweilige Komplexität wird anhand einer Ordinalskala von null bis fünf Punkten bewertet. Null bedeutet, dass die Einflussgröße irrelevant ist. Fünf Punkte bedeuten, dass die Einflussgröße sehr wichtig ist.
- Durch die Bewertung der einzelnen technischen Einflussgrößen T_i mit [0..5] Punkten erhält man den TCF über folgende Formel:

$$TCF = 0{,}58 + 0{,}01 * \sum_{i=1}^{13} (T_i * Wi)$$

Use Case Points (UCP)
Zählmaß Technische und Projektumgebungsfaktoren (iii)

- Analog zum TCF wird der EF durch acht Einflussgrößen bestimmt. EF bewertet damit die Rahmenbedingungen des Projekts:

E_i	Einflussgröße	Gewicht W_i
E_1	Familiar with Rational Unified Process	1,5
E_2	Part time workers	-1
E_3	Analyst Capability	0,5
E_4	Application experience	0,5
E_5	Object oriented experience	1
E_6	Motivation	1
E_7	Difficult programming language	-1
E_8	Stable requirements	2

- Durch die Bewertung der einzelnen Einflussgrößen E_i mit [0..5] Punkten erhält man den EF über folgende Formel:

$$EF = 1{,}405 - 0{,}03 * \sum_{i=1}^{8} (E_i * Wi)$$

Use Case Points (UCP)
Zählmaß Technische und Projektumgebungsfaktoren (iv)

- Die Adjusted Use Case Points (AUCP) werden wie folgt berechnet:

$$AUCP = UUCP * TCF * EF$$

- Durch Multiplikation mit einem unternehmensspezifischen Produktivitätsfaktor (PF) im erwarteten Wertebereich [20 – 35 Mitarbeiterstunden/UCP] ergibt sich daraus der geschätzte Projekt-Aufwand PE in Mitarbeiterstunden.

$$PE = AUCP * PF$$

Bewertung der UCP-Methode

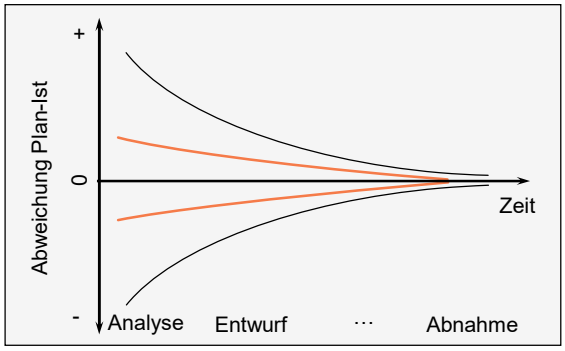
Es gibt eine Reihe von Kritikpunkten an der UCP-Methode nach Karner:

- Geringe Gewichtung der Akteure
- Mangelnde Praxiserfahrung
- TCF und EF nicht mehr zeitgemäß
- Die Komplexitätsstufen der TCF und EF zu vage
- Nicht für alle Entwicklungen geeignet, z. B. nicht für Anpassungen oder Wartungsprojekte
- Die verwendeten Skalentypen sind nicht definiert.

Vorgehen zur Aufwandsabschätzung

- Einzelschätzung:
  Schätzung der Arbeitspakete durch je einen Experten.
- Mehrfachbefragung:
  Getrennte Schätzung durch mehrere Experten und Mittelwertbildung ohne Informationsaustausch.
- Delphi-Befragung
  Getrennte Schätzung durch mehrere Experten mit anschließender Einigung auf einen Wert.
- Schätzklausur
  Mehrere Experten schätzen gemeinsam im Kollektiv.

Schätzungen sind immer unscharf



ohne Schätzmethoden

mit Schätzmethoden

ABER:

Schätzungen werden genauer durch

- mehr Informationen im Projektverlauf und
- Einsatz von systematischen Schätzmethoden.

Schätzwert > Zielwert? Was tun?

- **NEIN: Den Aufwand für alle Arbeitspakete pauschal reduzieren.**
- **NEIN: Qualitätssichernde Maßnahmen (Reviews, Tests, etc.) kürzen.**
- ☺ **JA: Prüfen, ob die Realisierung bestimmter Anforderungen in der nächsten Version erfolgen kann.**
- ☺ **JA: Eine Liste von Risiken erstellen, die beseitigt werden müssen, damit die geforderten Randbedingungen eingehalten werden können.**
- ☺ **JA: Rechtzeitige Erhöhung des Personals (Überstunden vermeiden).**

…

N

Timeboxing (i)
Planungstechnik im agilen Entwicklungsprozess

- "Verdrehte" Arbeitslogik: Nicht die Arbeitspakete werden geschätzt und mit Terminen versehen, sondern Termine werden festgesetzt zu den die Ergebnisse vorzulegen haben.
- Endtermin des Projekts wird also vorgegeben. Alles (Funktionalität, Qualität, Ressourcen, ...) muss sich an der Zeit orientieren.

Funktionsweise

- Für die Bearbeitung jedes Arbeitspakets wird eine fixe Zeitdauer (Timebox) geplant.
- Mit Ablauf dieser Zeit gilt das Arbeitspaket automatisch als erledigt. Der Verantwortliche muss sein Ergebnis präsentieren.
- Die Verantwortlichen erledigen die Aufgaben mit der bestmöglichen, in dieser Zeit machbaren Qualität und gehen dann zum nächsten Arbeitspaket über.

Timeboxing (ii)
Planungstechnik im agilen Entwicklungsprozess

Eigenschaften des Timeboxing:

- Im Projekt herrscht permanent ein Gefühl der Dringlichkeit.
- Der Fokus liegt auf das Generieren schneller Lösungen.
- Arbeitspakete werden in vorgegebener Zeit "erledigt".
- Qualität kann vernachlässigt werden.

Time Boxing eignet sich für eine Vielzahl von Projekten nicht.

Kapitel 3:
Planung von Softwareprojekten

3.1 Terminplanung

3.2 Methoden der Aufwandsschätzung

3.3 Kostenplanung

3.4 Ressourcenplanung

3.5 Inhalte eines Projektplans

Kostenplanung

Kostenplan eines Projekts stellt die Grundlage der Projekt-Budgetierung durch die Unternehmensleitung dar.

Hauptkostenfaktoren der SW-Entwicklung:

- Personalkosten
- Kosten der Fremdentwicklung

Weitere projektspezifische Kosten: Hardware, Werkzeuge, COTS-Komponenten (Commercial of the shelf), Testplattform, projektspezifische Weiterbildung, ...

Gemeinkosten (Kosten, die nur anteilig einem Projekt zugeordnet werden können): Mietkosten, Gehalt des Verwaltungspersonals, …

Kapitel 3:
Planung von Softwareprojekten

3.1 Terminplanung

3.2 Methoden der Aufwandsschätzung

3.3 Kostenplanung

3.4 Ressourcenplanung

3.5 Inhalte eines Projektplans

Ressourcenplanung

Ziel: Optimaler Ressourceneinsatz über die gesamte Projektlaufzeit.

In einem Softwareprojekt sind in aller Regel nur die Ressourcen zu planen, die im Entwicklungsprozess verwendet, aber nicht verbraucht werden (zumindest nicht kurzfristig): Personal, Infrastrukturen.

→ Im Weiteren wird nur die Personalplanung betrachtet.

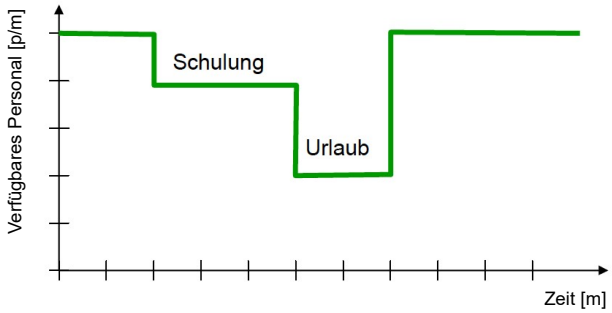
Bei der Personalplanung sind zu beachten:

- Qualifikation der Projektmitglieder
- Verfügbare Personalkapazität (zeitliche und örtliche Verfügbarkeit)

Ressourcenplanung
Vorgehen (i)

Schritt 1: Ermittlung der verfügbaren Personalressourcen (bzw. der Ist-Kapazität oder der möglichen Maximalauslastung)

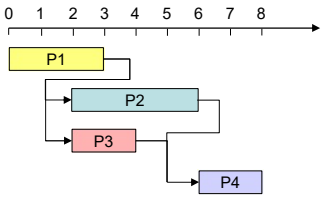
- Brutto-Zeitvorrat: Nominell verfügbare Arbeitszeit
- Netto-Zeitvorrat = Brutto-Zeitvorrat – (Urlaubszeit + Krankheitszeit + Schulungszeit + …)

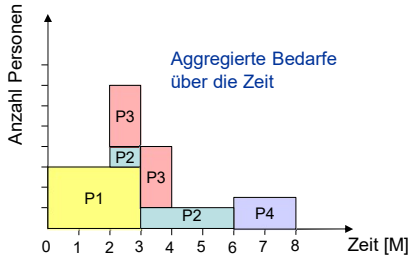


Ressourcenplanung
Vorgehen (ii)

Schritt 2: Errechnen des Personalbedarfs

APs	Aufwand [PM]	Dauer [M]
P1	9	3
P2	4	4
P3	6	2
P4	3	2

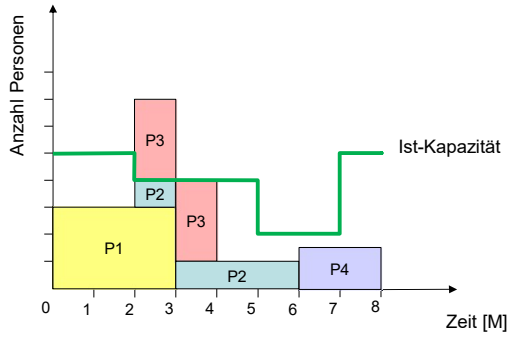




Zeit [M]

Ressourcenplanung
Vorgehen (iii)

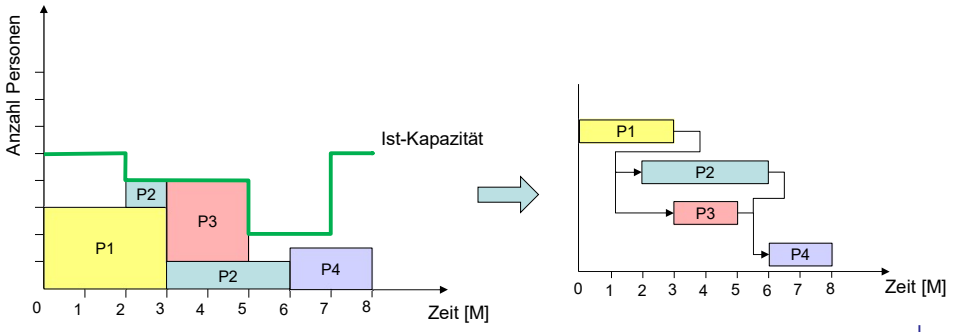
Schritt 3: Vergleich Vorrat und Bedarf (Vergleich Ist-Kapazität und Kapazitätsbedarf)



Ressourcenplanung
Vorgehen (iv)

Schritt 4: Optimierung der Auslastung (Änderung des Plans zwecks Übereinstimmung von Ist-Kapazität und Kapazitätsbedarf)

- Fall 1: Interne Optimierung möglich (Ausnutzen von Puffern)



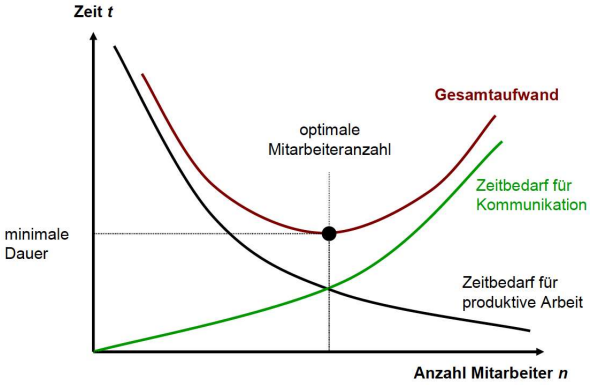
Ressourcenplanung
Vorgehen (v)

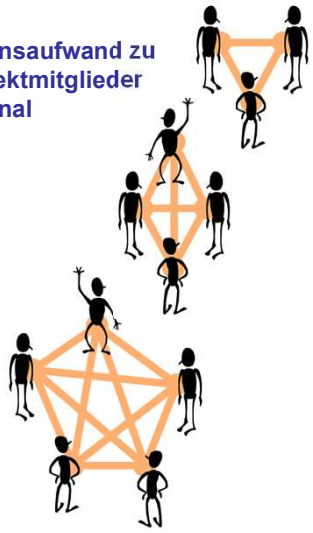
- Fall 2: Interne Optimierung nicht möglich. Hier gibt es folgende Handlungsweisen:
  - Ressourcen einkaufen oder geeignete Unteraufträge definieren und vergeben.
  - Fertigstellungstermin verschieben.
  - Kombination von beiden Maßnahmen.

Dieses Vorgehen der Ressourcenplanung ist praxisuntauglich. Warum?

Ressourcenplanung
Kommunikation und Personalbedarf (i)

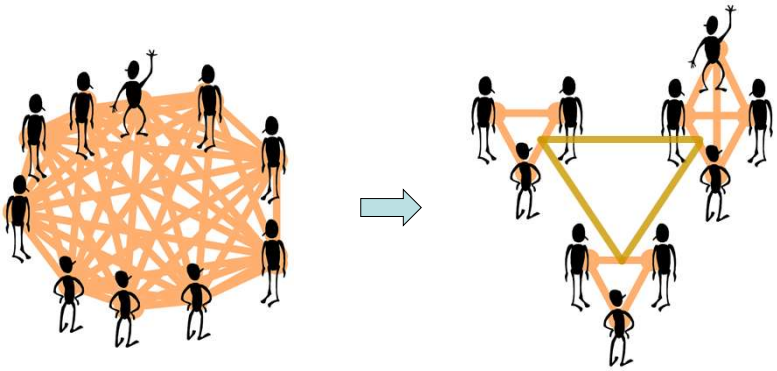
Bei der Ressourcenplanung ist der Kommunikationsaufwand zu berücksichtigen, da mit wachsender Zahl der Projektmitglieder der Abstimmungsaufwand im Team überproportional wächst .





Ressourcenplanung
Kommunikation und Personalbedarf (ii)

Das Zerlegen des Projekts in sinnvolle Teilprojekte reduziert den Abstimmungsaufwand.



Kapitel 3:
Planung von Softwareprojekten

3.1 Terminplanung
3.2 Methoden der Aufwandsschätzung
3.3 Kostenplanung
3.4 Ressourcenplanung
3.5 Inhalte eines Projektplans

Inhalte eines Projektmanagementplans nach IEEE 1058 (i)

Title Page
Change History
Preface
Table of contents
List of figures
List of tables

1. Project Overview

1.1 Purpose, Scope, and Objectives
1.2 Assumptions and Constraints
1.3 Project Deliverables
1.4 Schedule and Budget Summary
1.5 Evolution of the Plan

2. References

3. Definitions and Acronyms

4. Project Organization

4.1 External Interfaces
4.2 Internal Structure
4.3 Roles and Responsibilities

5. Managerial Process Plans

5.1 Start-up Plan
- 5.1.1 Estimation Plan
- 5.1.2 Staffing Plan
- 5.1.3 Resource Acquisition Plan
- 5.1.4 Project Staff Training Plan

5.2 Work Plan
- 5.2.1 Work Breakdown Structure
- 5.2.2 Schedule Allocation
- 5.2.3 Resource Allocation
- 5.2.4 Budget Allocation

Inhalte eines Projektmanagementplans nach IEEE 1058 (ii)

5.3 Project Tracking Plan
- 5.3.1 Requirements Management
- 5.3.2 Schedule Control
- 5.3.3 Budget Control
- 5.3.4 Quality Control
- 5.3.5 Reporting
- 5.3.6 Project Metrics

5.4 Risk Management Plan

5.5 Project Closeout Plan

6. Technical Process Plans

6.1 Process Model
6.2 Methods, Tools, and Techniques
6.3 Infrastructure
6.4 Product Acceptance

7. Supporting Process Plans

7.1 Configuration Management
7.2 Verification and Validation
7.3 Documentation
7.4 Quality Assurance
7.5 Reviews and Audits
7.6 Problem Resolution
7.7 Subcontractor Management
7.8 Process Improvement

8. Additional Plans

Annexes
Index

N

Strukturplanung am Beispiel eines Projekts zur Entwicklung eines Compilers

Was wird geliefert?

- Compiler-Software (Parser, Code-Generator, File System, Run Time System, GUI)
- Doku
- Installationssoftware

Welche Zwischenergebnisse?

Test-Werkzeug, Prototypen, Anforderungsspezifikation, Testspezifikation, …

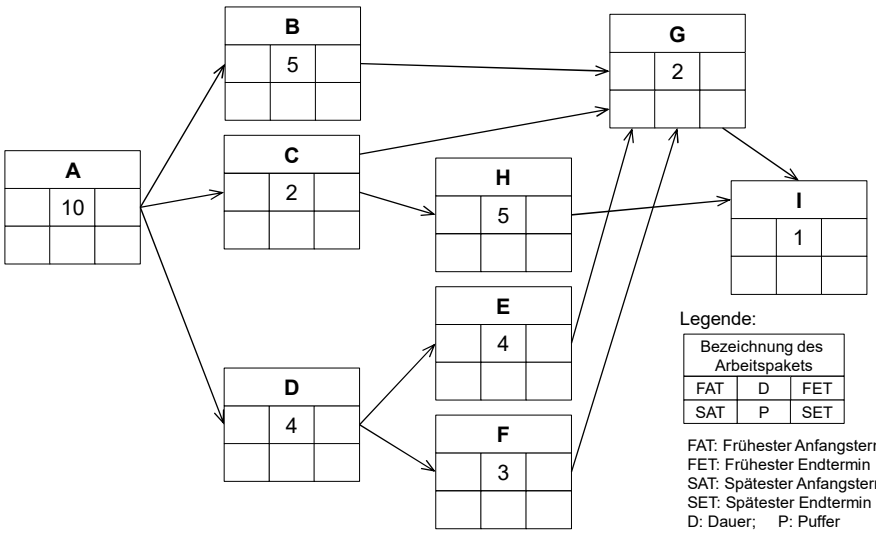
Welche Tätigkeiten?

PL, Konfigurationsmanagement, Testen (Durchführung), …

Terminplanung am Beispiel Projektierung einer Anlage (i)

Arbeitspakete		Dauer [Wochen]	Abhängig von
A	Entwurf	10	-
B	Fertigung der Zentraleinheit	5	A
C	Bereitstellung der Ein- und Ausgabegeräte	2	A
D	Projektierung der Software	4	A
E	Projektierung der Testprogramme	4	D
F	Erstellung kundenspezifischer Programme und Testprogramme	3	D
G	Anlagentest	2	B, C, E, F
H	Bereitstellung der Anschluss-Hardware	5	C
I	Installation	1	H, G

Terminplanung am Beispiel
Projektierung einer Anlage (ii)



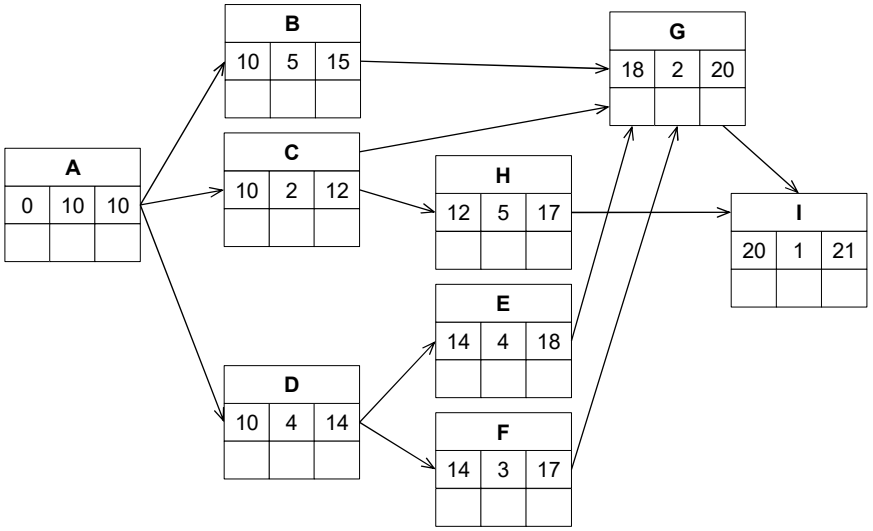
Legende:

Bezeichnung des Arbeitspakets		
FAT	D	FET
SAT	P	SET

FAT: Frühester Anfangstermin
FET: Frühester Endtermin
SAT: Spätester Anfangstermin
SET: Spätester Endtermin
D: Dauer; P: Puffer

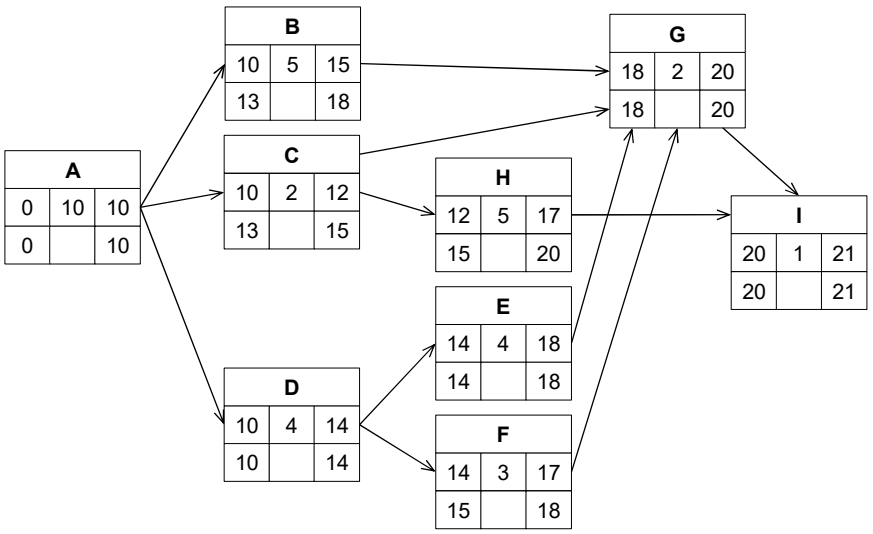
© Prof. Dr. M. Rezagholi, 2022

Terminplanung am Beispiel
Projektierung einer Anlage (ii) – FAT, FET



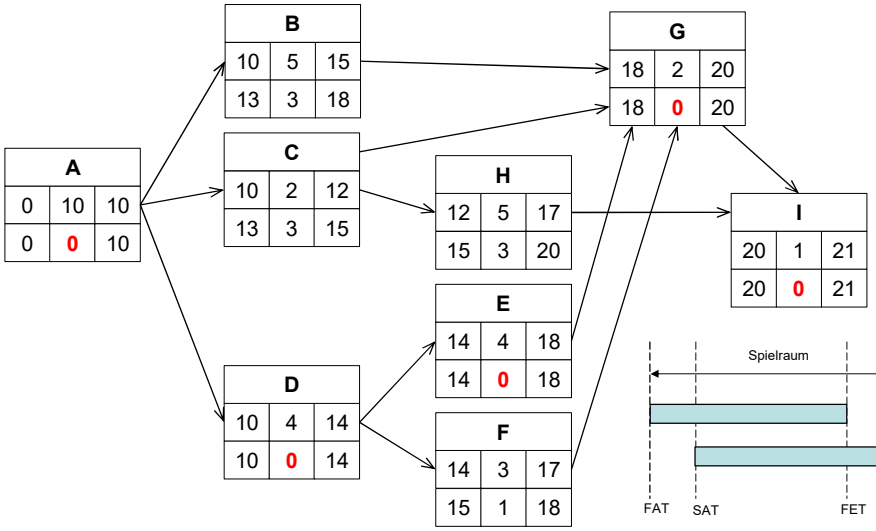
© Prof. Dr. M. Rezagholi, 2022

Terminplanung am Beispiel
Projektierung einer Anlage (ii) – SET, SAT



© Prof. Dr. M. Rezagholi, 2022

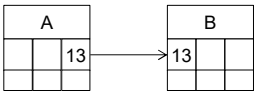
Terminplanung am Beispiel
Projektierung einer Anlage (ii) – Puffer



© Prof. Dr. M. Rezagholi, 2022

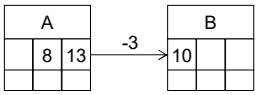
Gängige Abhängigkeiten zwischen zwei Arbeitspaketen A und B

Normalfolge (Ende-Anfangs-Folge): Anfang des Arbeitspakets B hängt vom Ende des Arbeitspakets A ab.



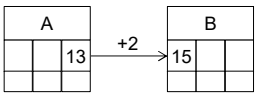
Überlappung:

Arbeitspaket B Beginnt vorm Ende des Arbeitspakets A



Verzögerung:

Arbeitspaket B beginnt nicht unmittelbar nach dem Ende des Arbeitspakets A.



COCOMO II Grundformel

(Optimale) Projektdauer (Projektlaufzeit, Entwicklungszeit)

$$\text{Projektdauer [M]} = 2{,}5 * (\text{Aufwand[PM]})^{S}$$

s = 0,4 für Standardsoftware
s = 0,38 für Stapelverarbeitungssysteme (Batchsysteme);
s = 0,35 für Dialogsysteme/Onlinesysteme;
s = 0,32 für Echtzeitsysteme

Teamgröße

$$Teamgröße[P] = \frac{Aufwand\ [PM]}{Projektdauer\ [M]}$$

Beispiel: Motorsteuerung

Aufwand [PM]: 22

Dauer [M] = 2,5 * 22^0,32 = 6,7

Teamgröße [P] = 22 / 6,7 = 3,3

Legende: M: Monat; PM: Personenmonat; P: Anzahl Projektmitarbeiter; KDSI: Kilo Delivered Instructions